Message-handling systems based on the CCITT X.400 recommendations

by T. E. Schütt
J. B. Staton III
W. F. Racke

Message-handling systems allow the exchange of electronic mail between computers. The International Telegraph and Telephone Consultative Committee (CCITT) has proposed a standard for message-handling systems in the form of the X.400 series of recommendations that has been widely recognized by computer manufacturers and communications carriers. This paper provides a tutorial on the X.400 recommendations and then describes two prototypes developed by the IBM European Networking Center in Heidelberg, Germany, in cooperation with its research partners. The prototypes were demonstrated together with X.400 prototypes from other manufacturers at the CeBIT 86 trade fair in Hannover, Germany.

Application programs that cooperate with one another to perform the exchange of messages between computers or message-handling systems have been in use since the late 1960s. Many examples of such applications were constructed to provide message exchange or electronic mail within networks of computers offered by the same manufacturer. In IBM, examples of product offerings that include message-handling functions are the Distributed Office Support System (DISOSS) and the Professional Office System (PROFS).[1] These products were constructed for exchanging messages within networks using Systems Network Architecture (SNA) and Remote Spooling Communication Subsystem (RSCS) protocols under the Virtual Machine (VM) operating system, respectively. Another example is the mail transfer system in the Department of Defense Advance Research Projects Agency Network (ARPANET).

During the period from 1980 to 1984, Study Group VII of the International Telegraph and Telephone Consultative Committee (CCITT) developed a set of recommendations for message-handling systems[2] that allow computers of different manufacturers to exchange messages. The recommendations utilize the communications model and protocols that have been developed by the International Standards Organization (ISO), while proposing new protocols to perform the message-handling application. In the fall of 1983, a reasonably complete set of eight new recommendations was released. These recommendations, approved by the CCITT in October 1984, are known collectively as the X.400 recommendations on message-handling systems (henceforth referred to simply as X.400).

Since 1984, there have been a number of efforts by associations of manufacturers, public telephone and telegraph associations, and government-sponsored standards agencies[3] directed at identifying reasonable subsets of function that can be implemented as well as providing guidance to implementers on ambiguities in the recommendations. These efforts have resulted in the definition of a number of X.400 functional standards or subsets of the X.400 protocols that can be implemented to allow different computers to exchange messages. Two key functional standards have been defined for X.400 that are often referred to as the CEPT and CEN/CENELEC profiles. The CEPT

© **Copyright** 1987 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

functional standard has been defined for exchanging messages between private companies and message-handling systems operated by public telephone and telegraph companies (PTTs), and the CEN/CENELEC functional standard has been defined to describe the interfaces of message-handling systems operated by private companies.

A number of manufacturers in Europe, including IBM, began to demonstrate prototype message-handling systems based on the X.400 recommendations in 1985. In 1986, IBM participated with other manufacturers and nonprofit institutions to demonstrate the exchange of X.400 messages at the CeBIT trade fair in Hannover.[4]

The X.400 recommendations are important for several reasons. They are one of the first application layer standards (layer 7) of the ISO reference model for open systems interconnection,[5] and, as such, they will influence the development of the services and protocols for other application layer standards.[6] In fact, the X.409 recommendation on Presentation Transfer Syntax and Notation, which contains the description of the representational techniques used to specify and encode messages, has set the direction for the Abstract Syntax Notation 1 (ASN.1). This notation is currently being defined in ISO to be used to specify the data units of other application layer protocols.

The X.400 recommendations address an important application area for most users, that is, the area of electronic messaging in a heterogeneous environment, and thus there is a high degree of user interest in progress toward their implementation. Furthermore, other groups standardizing specific types of message transfer protocols, such as the ISO TC68 committee on Banking and Related Financial Services, which is responsible for the standardization of electronic funds transfer protocols, will also be strongly influenced by the X.400 recommendations.

Finally, the availability of X.400 services through the public networks worldwide, as has already been announced in many countries in Europe, also contributes to its importance as an international standard.

Major concepts and terminology

The X.400 recommendations for message-handling systems[2] consist of eight recommendations that together define the services and protocols for message exchange in open systems. The eight recommendations are the following:

- X.400 System model-service elements
- X.401 Basic service elements and optional user facilities
- X.408 Encoded-information-type conversion rules
- X.409 Presentation transfer syntax and notation
- X.410 Remote operations and reliable transfer server
- X.411 Message transfer layer
- X.420 Interpersonal messaging user agent layer
- X.430 Access protocol for Teletex terminals

The first recommendation, X.400, provides an overview of the message-handling system (MHS) model. In this model, a *user* is a person or an application program that sends or receives messages. When sending a message, the user is referred to as the *originator* of a message. When receiving a message, the user is referred to as a *recipient* of a message. The user prepares and receives messages through the assistance of a *User Agent* (UA), which is an application process that interfaces with the *Message Transfer System* (MTS) to submit and receive messages on the user's behalf. The message transfer system is the set of *Message Transfer Agents* (MTAs) that perform such functions as relaying the messages to the appropriate destinations and providing safe storage for messages in transit.

User agents could be such applications as electronic funds transfer or services to interconnect university libraries.

As application entities in the ISO reference model, the message transfer agent makes use of the *Reliable Transfer Server* (RTS) to establish session connections and reliably transfer messages through these connections on behalf of the message transfer agent. If a connection becomes inactive, it is up to the RTS to re-establish the session and to continue the transfer

of the message until the transfer is complete. The specific user agent described in the X.420 recommendation is the *Interpersonal Messaging* (IPM) user agent. This user agent provides for the transfer of office mail to the message transfer system. Other user agents could be defined for such applications as electronic funds transfer in the banking industry or library services to interconnect university libraries.

In the physical mapping of the message-handling system, the user agent can reside in the same processing system as the message transfer agent being accessed by I/O devices such as terminals attached to the processing system. The user agent can also reside in an intelligent workstation or processor that is separate from the processing system containing the message transfer agent. This is made possible by the definition in the X.400 recommendations of a *submission and delivery entity* (SDE), which submits and receives messages from the MTA on the basis of a defined interface and specific SDE-MTA protocol. Organizationally, the message transfer agents are grouped into *administration management domains* (ADMDs) and *private management domains* (PRMDs). The administration management domains are the responsibility of the local network provider or national carrier, whereas a private management domain is maintained by a private organization or company. In the X.400 recommendations, it was envisioned that private management domains would exchange messages with each other only, by first sending them through administration management domains and thus using the local carriers to route the messages. Whether this actually occurs will depend upon the services provided by the various national carriers and may vary from country to country. The form of the message itself, as it is transferred from originator to recipient, consists of an *envelope* and *content*, which are analogous to an envelope and letter sent through the postal system. The envelope contains the addresses of the recipients and the originator, in addition to other information such as whether the originator requires confirmation that the message has been delivered.

Actually, three envelopes are created during the entire process of delivering a message. The user agent submits the message to the MTA by transferring the content plus a submission envelope. The message is transferred from one MTA to another by means of a relaying envelope. The final MTA transfers the message to the recipient user agent by means of a delivery envelope.

The content of the message consists of two parts, *heading* and *body*. The heading is analogous to the heading of an interoffice memo and contains information such as the originator, recipient, unique message identification, subject, and references to other messages. The body of the message is the actual information that the originator wishes to convey to the recipient. It is classified by body type, which describes the encoding scheme used for the information. Examples of currently defined body types include International Alphabet 5 (IA5) Text, Teletex, and Group 3 Facsimile (G3Fax).

The originator and recipient(s) of a message are identified by an *Originator/Recipient name* (O/R name). The O/R name may be supported in one of two basic forms. In the first form, the O/R name consists of a subset of the set of attributes as follows:

- Country name
- Administration domain name
- Private domain name
- Personal name
- Organization name
- Organizational unit names
- Domain-defined attributes

In the second form, the O/R name consists simply of the X.121 address and, optionally, a telematic terminal identifier. Because the O/R name is also used for routing, it also serves as the address of the originator and recipient.

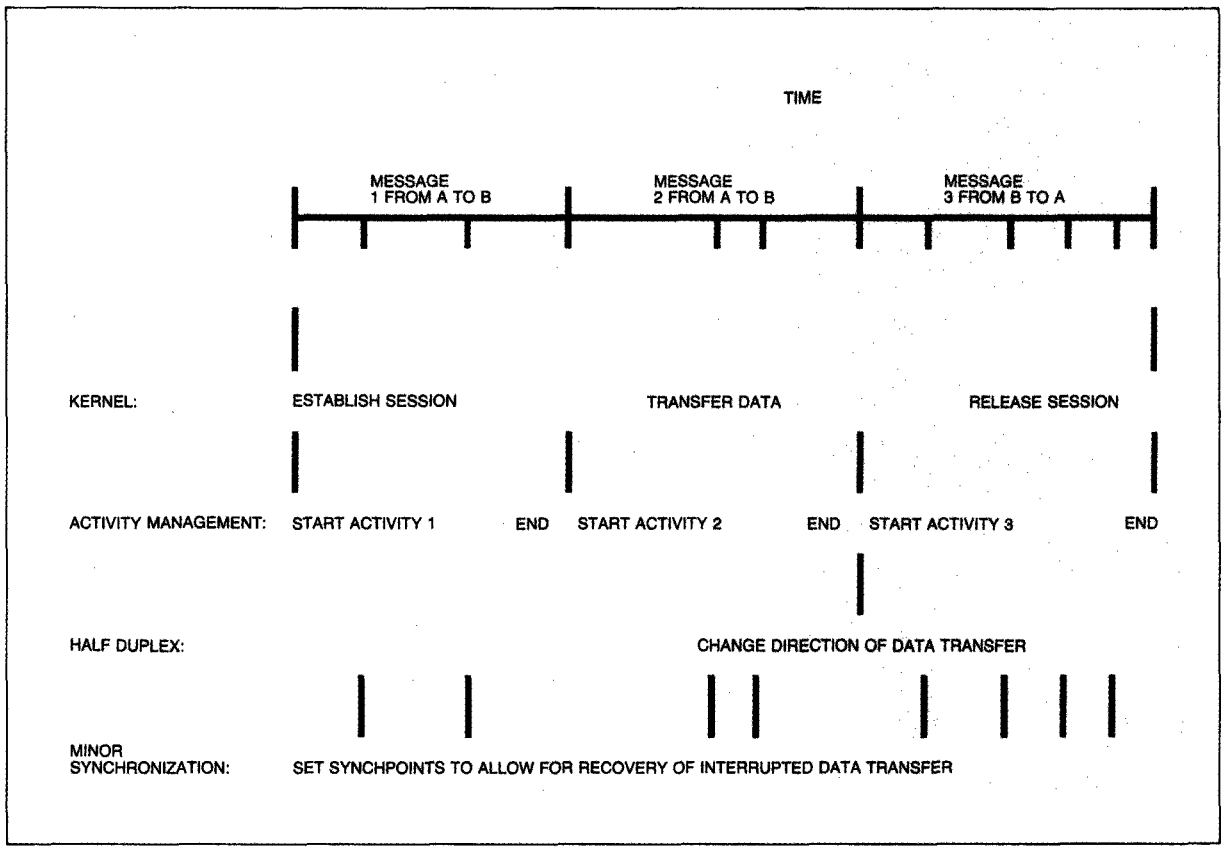
For migration purposes, a protocol is defined in the X.430 recommendation to allow Teletex terminals to access the message-handling system through the use of a special user agent known as a Teletex access unit (TTXAU).

Relationship to ISO standards

To ensure widespread acceptance and to expedite the implementation of the X.400 recommendations, the CCITT decided to build upon existing ISO or CCITT communication standards. Draft standards already existed in 1984 for the lower five layers of the OSI reference model, and are simply referenced by the X.400 recommendations for message-handling systems. Only the two highest layers (application and presentation layers) required specification by the CCITT.

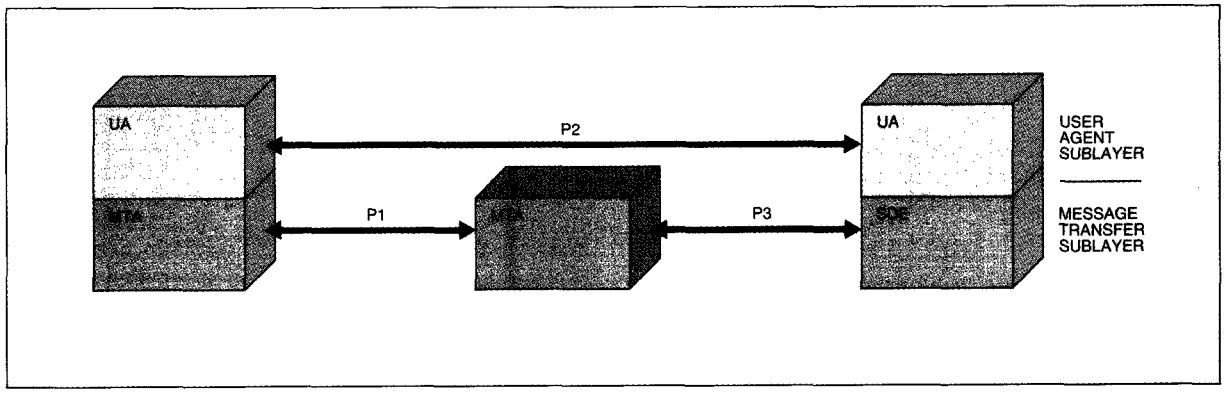
Although the presentation layer is formally left empty by the X.400 model of a message-handling

Figure 1 Reliable Transfer Server (RTS) usage of the session functional units



system, the CCITT accepted a separate recommendation (X.409) to specify the encoding rules for messages and control information to be used by the subcomponents of the application layer. The application layer is logically split into two sublayers, the lower one consisting of two separate components. These three subcomponents are described by the following three new recommendations: X.420 for the user agents (UAs); X.411 for the message transfer agents (MTAs); and X.410 for the reliable transfer server (RTS), a separable subcomponent of the MTA. The RTS, as the lowest newly defined subcomponent of the application layer, makes direct use of a subset of the ISO session services[7] that is commonly known as the *Basic Activity Subset* (BAS). The session functional units of the Basic Activity Subset that are required by the RTS consist of the session kernel, exceptions, activity management, half-duplex, and minor synchronization. These functional units allow the RTS to provide for a reliable and recoverable transfer of individual messages between cooperating MTAs. An example of the usage of the session functions by the RTS is given in Figure 1. The session connections that are operated by the RTS to do the message transfer are based on transport connections that can be established and maintained via the transport services defined by the respective ISO[8] and CCITT specifications.[9] The current version of the recommendations requires only the functions of the ISO transport protocol class 0 as a mandatory feature of an MHS implementation. Other classes of transport layer protocols, such as the widely discussed ISO transport protocol class 4, have been left for further study.[10,11] The class 0 transport connections may be based on any suitable network service. The choice of the protocols for the lower three ISO layers (network layer, data link layer, and physical layer) is explicitly left to the network providers. Only the procedures for network layer gateways between different provider choices (e.g., packet-switched versus circuit-switched networks) are specified in a companion document.[12] Because X.25 is a standard service provided by PTTs throughout Europe, it will be the

Figure 2 Sublayers and protocols of an interpersonal messaging system



choice of the lower layers for most European implementations of X.400.

We envision that in the future other transport protocol classes will be incorporated into the CCITT X.400 recommendations and also into the ISO Message Oriented Text Interchange System (MOTIS) draft standards.[13–16] Such enhancements could allow for a standardized usage of leased lines, local-area networks (LANs), or Integrated Services Digital Networks (ISDNs).

The X.400 recommendations

The X.400 message transfer agent, submission and delivery entity, user agent, and the corresponding peer protocols for communication among the components reside at the application layer of the ISO Reference Model. The application layer is divided into the following two sublayers:

- Message Transfer Layer (MTL), which contains the functions of the message transfer system as provided by the message transfer agents and submission and delivery entities
- User Agent Layer (UAL), which contains the functions provided by the user agents

This layered representation of the X.400 model for message-handling systems allows, in a manner similar to the layered OSI Reference Model, the identification of unique entities in each of the two sublayers, the identification of the protocols used between peer entities, and the independent specification of the service interfaces for these entities. For the specific case of the Interpersonal Messaging System, the different entities and their peer protocols are shown in Figure 2.

Message transfer layer. The X.411 recommendation entitled Message-Handling Systems: Message Transfer Layer contains the specification of the message transfer services including the specification of the P1 and P3 protocols. The P1 protocol is used for communicating between MTAs, and the P3 protocol is used between an MTA and an SDE. The services of the message transfer layer provide the user agent with the means for transferring messages to and from the message transfer system. The services can be requested by a user agent from its message transfer agent either directly, when both are located in the same processing system, or through the use of a submission and delivery entity, in the case of a remote user agent. In either case the services provided to the user agent are the same.

Message transfer services. The service interface between the message transfer layer and the user agent is described by a set of thirteen *service primitives.* (See Table 1.) The use of service primitives for the description of a service interface is an abstract way of capturing only those details of the interaction between two adjacent entities that are required for the layer service. Parameters are associated with a service primitive when it is necessary to transfer additional information between entities. A service primitive neither specifies nor constrains the implementation of the entities or the service interface between them.[17]

The LOGON and LOGOFF service primitives establish and release a dialogue between the user agent and the message transfer agent. The establishment of a dialogue can be initiated either by the user agent via (UAL)LOGON, or by the message transfer layer via (MTL)LOGON. Only the user agent is able to release a

Table 1 Service primitives of the message transfer layer

(UAL)LOGON	User-initiated access establishment service
(MTL)LOGON	MTL-initiated access establishment service
LOGOFF	Access termination service
(UAL)CHANGE-PASSWORD	Password change service
(MTL)CHANGE-PASSWORD	Password change service
REGISTER	Registration service
(UAL)CONTROL	Hold for delivery service
(MTL)CONTROL	Restriction indication service
SUBMIT	Message submission service
CANCEL	Cancel deferred delivery service
PROBE	Probe service
NOTIFY	Message notification service
DELIVER	Message delivery service

dialogue using LOGOFF. The LOGON process is secured in either case by a password. Both the user agent and the message transfer layer passwords can be changed with the (UAL)CHANGE-PASSWORD and (MTL)CHANGE-PASSWORD primitives, respectively.

Each message transfer agent maintains information concerning the capabilities of the user agents that are served by the MTA, such as message types supported and maximum length of deliverable messages. A user agent can modify these values by issuing the REGISTER service primitive with appropriate parameters.

Whereas modifications made with the REGISTER service primitive remain in effect until the next REGISTER is issued, the (UAL)CONTROL and (MTL)CONTROL service primitives are used to modify parameters only for the duration of an established dialogue, that is, between LOGON and LOGOFF. For example, the two service primitives may be used to control congestion between an MTA and a UA by halting the submission or the delivery of further messages for a period of time.

The SUBMIT primitive initiates the transfer of a message to one or more recipients. The parameters of SUBMIT are the content of the message to be transferred and the submission envelope. The submission envelope contains the O/R-name(s) of the recipient(s), the originator O/R-name, the priority of the message, and the request for Delivery Notification(s). If requested, the message transfer layer uses a delivery notification to inform the originating user agent about the successful delivery of the message to the recipient user agent; that is, the MTS has put the message into the *mailbox* of the recipient. A nondelivery notification informs the user agent (UA) that the message transfer system was unable to deliver the message to the recipient user agent. For example, the recipient UA may be unknown to the message transfer system. One parameter of SUBMIT allows the specification of a deferred delivery time before which the message must not be delivered by the message transfer system.

Figure 3 Structure of a user message

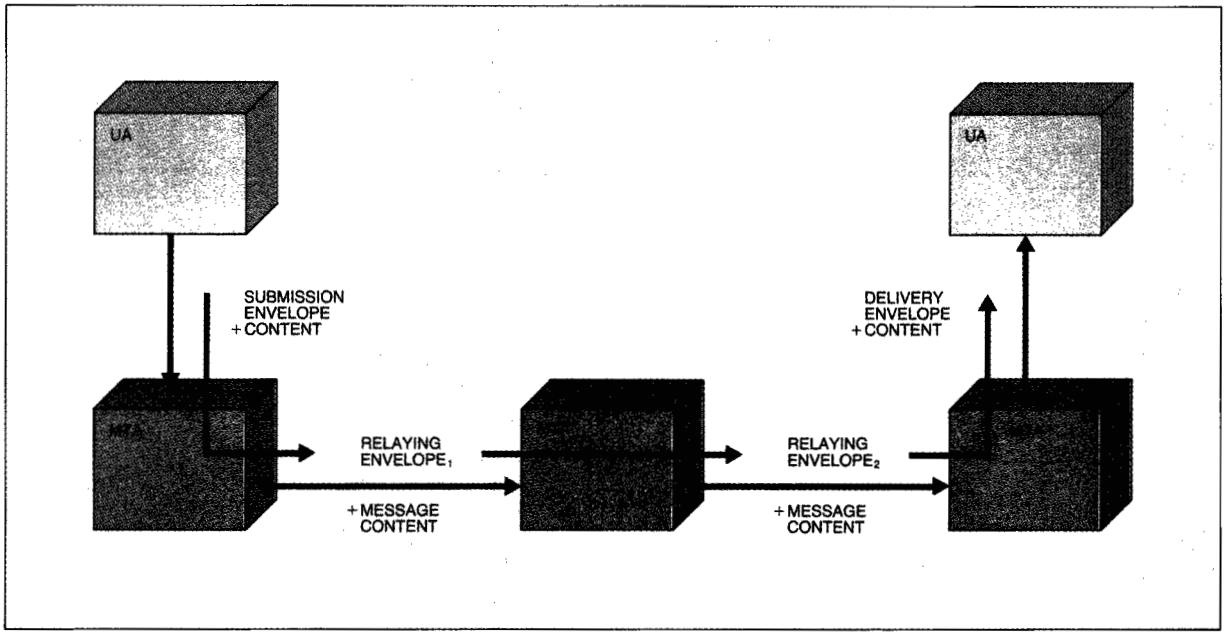
The CANCEL service primitive enables the originating user agent to cancel a previously submitted message with a specified deferred delivery time. However, this is possible only if the CANCEL is received by the message transfer agent responsible for the submitting user agent before the message is forwarded to another message transfer agent.

A user agent issues the PROBE service primitive to determine whether the path to a recipient is operational. This is a procedure that may be needed in problem determination.

The message transfer layer uses the NOTIFY service primitive to inform the user agent about the delivery or nondelivery of a previously submitted message or to convey the results of a PROBE. The NOTIFY parameters contain additional information for the user agent, such as the delivery time of a message or the reason for its nondelivery.

The message transfer layer issues the DELIVER service primitive to deliver a message to a recipient user agent. The parameters of DELIVER are the content of the delivered message and the delivery envelope with the originator O/R-name, the recipient O/R-name, optionally the O/R-names of other recipients, the priority of the message, and the message submission time.

Figure 4 Envelopes involved in the submission, relaying, and delivery of a message



Message transfer agents and the P1 protocol. Some of the service primitives described in the previous section are provided locally in a single message transfer agent and do not depend upon communication between MTAs. These include LOGON, LOGOFF, and REGISTER. The other service elements, such as SUBMIT and PROBE, require the cooperation of message transfer agents, which is achieved by means of the message transfer, or P1 protocol.

The P1 protocol transfers message protocol data units (MPDUs) of three types. These are a user MPDU carrying a message from one message transfer agent to another, a delivery report MPDU to transfer delivery or nondelivery information, and a probe MPDU.

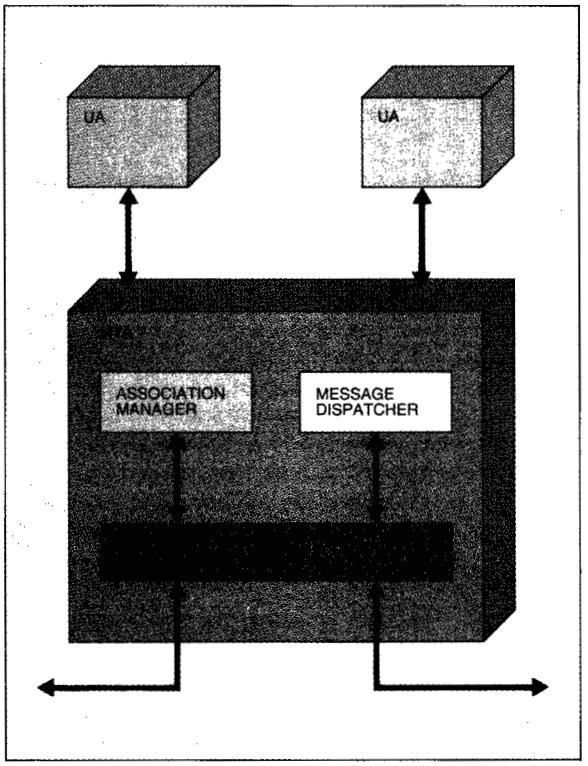
The user MPDU consists of two parts: the relaying envelope and the message content, as shown in Figure 3. The relaying envelope contains the information necessary for the cooperation of message transfer agents for relaying the message. The envelope is constructed by the MTA serving the originating user agent from the information contained in the submission envelope. The following are the basic fields of the relaying envelope:

- MPDU identifier
- Originator O/R-name
- Recipient O/R-name
- Priority
- Deferred delivery time
- Trace information

The relaying envelope is encoded in a bit string according to the encoding rules of the X.409 recommendation. The encoded envelope and the message content are then transferred to the next MTA. The message content is transparent to the message transfer system. While being transferred through the message transfer system, certain fields of the relaying envelope are modified to reflect the status of the message. An example is the trace information that is added by each message transfer agent through which the message passes. The MTA that is serving the recipient user agent uses the relaying envelope to create the delivery envelope. The delivery envelope together with the message content is passed to the recipient user agent by means of the DELIVER service primitive. The different envelopes and their relationship are depicted in Figure 4.

A further refinement of the MTA into its three subcomponents illustrates in more detail the functions

Figure 5 Associations with adjacent Message Transfer Agents (MTAs)



performed by an MTA when interacting with other MTAs using the P1 protocol. These three subcomponents are the Message Dispatcher, the Association Manager, and the Reliable Transfer Server (RTS). The relationship of these components within the MTA is shown in Figure 5.

The message dispatcher processes the P1 protocol and is driven by MPDUs received from other message transfer agents and by messages or probes initiated by the user agents of the MTA. One of the functions performed by the message dispatcher is the generation of the Delivery Report MPDUs. When relaying a message the message dispatcher uses the recipient O/R-name to determine the routing and address information for forwarding the message to the next message transfer agent(s). Copies of the message are created and transferred to different MTAs in the event that the recipients of the message are reached via different paths within the message transfer system.

Table 2 Service primitives of the Reliable Transfer Server (RTS)

OPEN	Establishment of an association
CLOSE	Release of an association
TURN-PLEASE	Request for exchange of the turn
TURN-GIVE	Exchange of the turn
TRANSFER	Reliable transfer of an application protocol data unit
EXCEPTION	Indication of transfer failure

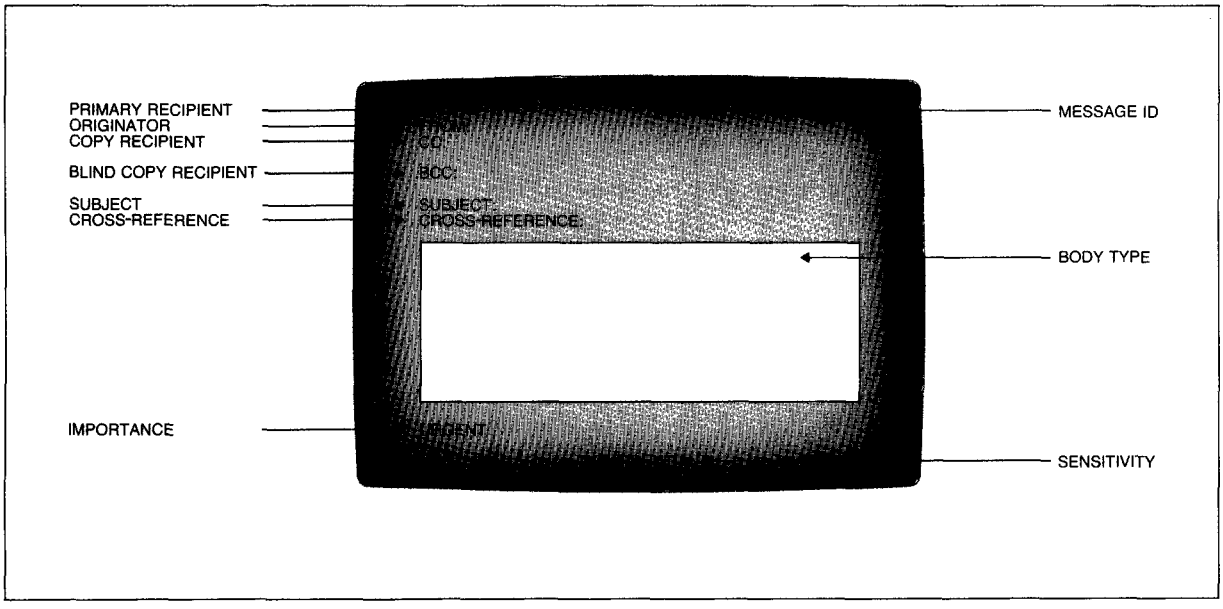
The message dispatcher transfers an MPDU to an adjacent message transfer agent in a single transaction over an association, which is a logical relationship between peer entities for the exchange of protocol data units. As defined in the OSI Reference Model, associations are realized through connections of the next lower layer. Both the association manager and the reliable transfer server provide the functions to support the message dispatcher's single-transaction view of the message transfer. The association manager initiates and controls the establishment and the release of associations, whereas the reliable transfer server is responsible for providing associations and for completely and reliably transferring MPDUs by means of them.

The RTS provides the MTA with a simplified interface to the session layer that allows an MPDU to be transferred in a single transaction. A set of six service primitives for the reliable transfer server is described in the X.410 recommendation, as shown in Table 2.

The OPEN and CLOSE service primitives are used respectively by the message transfer agent to initiate the establishment of an association for the transfer of Application Protocol Data Units (APDUs) or to release an existing association. The reliable transfer server provides two-way-alternate (half-duplex) associations. TURN-PLEASE and TURN-GIVE request and exchange the right to send data over an association. With the TRANSFER service primitive, the reliable transfer of data is requested. The reliable transfer server informs the MTA by way of the EXCEPTION primitive in the event that it cannot complete the requested transfer of data.

Submission and delivery entities and the P3 protocol. User agents request the services of the message transfer layer through the message transfer agent that serves the UAs. In the case of a remote user agent, where the UA and the MTA reside in different processing systems, a special way to interconnect them

Figure 6 Interpersonal Message (IPM)



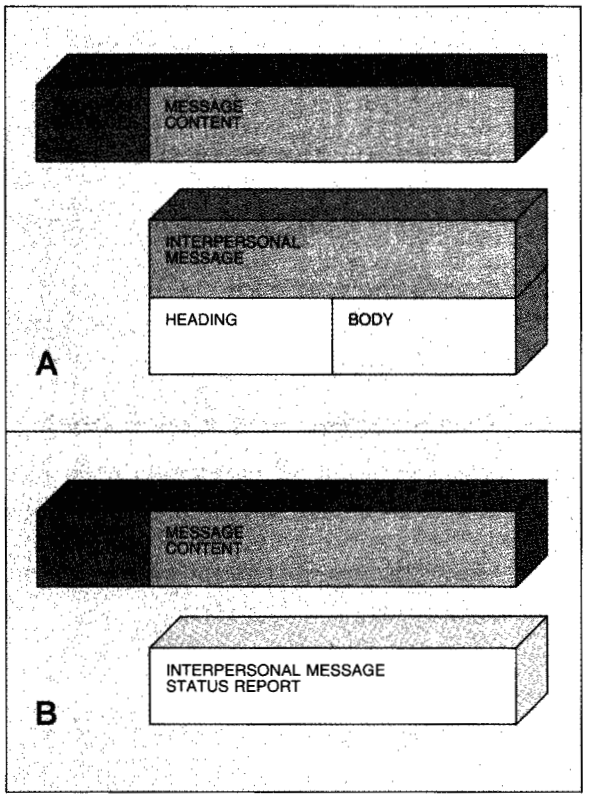
has to be introduced. User agents and message transfer agents belong to different sublayers of the application layer and are not peer entities. Therefore, a direct communication between them in the OSI environment is not possible. The Submission and Delivery Entity (SDE) belonging to the message transfer layer solves this problem. The SDE interacts with its peer message transfer agent to provide the message transfer service to the remote user agent using the P3 protocol. Whereas the relaying of messages between message transfer agents by means of the P1 protocol is based on a store-and-forward technique, the communication between an MTA and an SDE is interactive and transaction-oriented. The P3 protocol reflects this communication. The remote operations macro, specified in the X.410 recommendation, provides the means for the remote invocation of an operation and the transfer of the required arguments for that operation, the return of the results when the operation has completed successfully, and the return of an error report in the event that the operation fails. The X.411 recommendation specifies a protocol for remotely invoking the service primitives that comprise the MTA-UA interface.

The submission and delivery entity has the task of transferring the service primitives of the message transfer layer and their arguments between a user agent and its message transfer agent in both directions as defined in the P3 protocol. Every service primitive together with its parameters is mapped into one Operation Protocol Data Unit (OPDU). The invocation mechanism provided through the remote operations facility is used to request the remote invocation of that service primitive. Return codes and errors from the execution of the service primitives are treated in the same way.

Interpersonal Messaging user agent layer. The Interpersonal Messaging (IPM) user agent sublayer is described in the X.420 recommendation, entitled Message Handling Systems—Interpersonal Messaging User Agent Layer. This recommendation defines the IPM user agent and the P2 protocol for peer communication. The IPM services are provided to the users through a standard application user interface, such as an editor.

Interpersonal Messaging services. The IPM user agent supports a number of functions or service elements similar to those used in the typical office memo in Figure 6. These include the originator of the message, primary and copy recipients, blind copy recipients, the subject of the message, the importance and the sensitivity, a cross-reference, and a message identifier for future reference.

Figure 7 (A) Interpersonal Message (IPM); (B) IPM status report



The content of the message is referred to as the message body; it can consist of multiple body parts, each of which may have a different body type or character set and media. The body type is dependent upon the character set and media used to transmit the message. Examples of body types are IA5 (ASCII) text, Teletex document, and G3Fax. Due to potential differences in the capabilities of the originator and recipient user agents, the body may undergo conversions, so that the body type of a body part may change during the transfer of the message.

Interpersonal Messaging user agents and the P2 protocol. IPM user agents interact by means of the P2 protocol. Two types of protocol data units are specified for this protocol; the IPM protocol data unit and the Status Report protocol data unit. The IPM protocol data units carry interpersonal messages generated by the originator and transferred to the recipients. A Status Report protocol data unit contains information for the originating user agent about the transfer of the message to the recipient user agent(s). Both types of protocol data units are transferred between the user agents by the message transfer system. They form the transparent content of a P1 message, as illustrated schematically in Figure 7.

The IPM protocol data unit contains both the heading and the body of the interpersonal message. The heading fields of an IPM protocol data unit are the following:

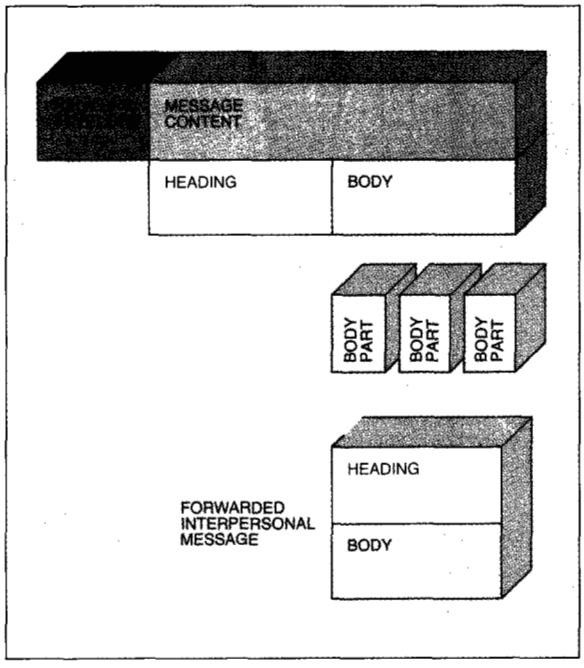
- IPM message identification
- Authorizing user
- Originator
- Primary recipients
- Copy recipients
- Blind copy recipients
- In reply to
- Cross-references
- Obsoletes
- Subject
- Importance
- Sensitivity
- Reply by (time)
- Reply to users
- Expiration date
- Autoforwarded

These heading fields reflect all of the services offered by the IPM user agent layer to the user when creating an interpersonal message. They characterize the message for the recipient(s) and will be indicated to them by the recipient user agent(s). Besides the delivery and nondelivery notifications of the message transfer service, an originator of a message can request a receipt or a nonreceipt notification from the IPM user agent layer. These notifications are used to inform the originator that an interpersonal message was or was not received by the intended recipient. The body of an interpersonal message may consist of different body parts, as shown in Figure 8. A forwarded interpersonal message contains a complete IPM protocol data unit in its body.

Before submitting an interpersonal message or status report to the message transfer system, the protocol data units are encoded by the originating user agent according to the encoding rules defined in the X.409 recommendation.

The P2 protocol definition also specifies the operations an IPM user agent is required to perform when interacting with other IPM user agents. Moreover,

Figure 8 Body and body parts of an Interpersonal Message



rules are provided that specify how an IPM user agent has to use the services of the message transfer layer. One specific operation of a user agent relates to the use of the blind copy recipient service. When a user has specified blind copy recipients for a message, the user agent has to generate two different IPM protocol data units. One is the copy for the blind copy recipients and contains the blind copy recipient heading field, whereas the other does not. This ensures that the primary and copy recipients of the message receive no knowledge that a blind copy of the message was sent to other recipients.

An X.400 prototype for the VM/SP operating system

On the basis of the CCITT X.400 series of recommendations and the specifications for realization of the Deutsches Forschungsnetz (DFN) message-handling system,[18] two X.400 prototype systems using the VM/SP operating system were developed at the IBM European Networking Center (ENC) in Heidelberg. The first is a native X.400 system using the IBM office system product PROFS as a user agent, and the second is an X.400 gateway between RSCS-based networks,[19] such as the IBM VNET, EARN, or BITNET;[20] and X.400 networks, such as the DFN (Germany) or OSIRIDE (Italy).[21] Both prototypes were developed as joint projects by the European Networking Center, the Gesellschaft für Mathematik und Datenverarbeitung (GMD), which is a German-government-sponsored research agency, and Queen's University in Kingston, Ontario, Canada.

PROFS/X.400 prototype system. In the PROFS/X.400 prototype, the IBM Professional Office System (PROFS) program product provides the interface to the user for composing, sending, and receiving messages. No modifications to PROFS were required. Thus the prototype allows a PROFS user to communicate transparently with other X.400 systems without learning new commands or adapting to a new operational environment.

The ability to create and transmit PROFS notes to one or more X.400 recipients is provided. To support this function, it was necessary to develop a bridge from PROFS to X.400 that converts the PROFS-note format to the interpersonal messaging format defined in the X.420 recommendation, and to create the necessary envelope information required by the X.411 recommendation for the message transfer layer. The whole message and its envelope are encoded according to the X.409 recommendation and then passed to the MTA for distribution.

The MTA employs a local directory service to map the (logical) O/R names of the intended recipient(s) of the message to their physical address(es), i.e., the ordered triple X.25 DTE address, transport service access point address, and session service access point address. This physical destination address identifies an adjacent MTA to which messages for a particular recipient are to be forwarded. The adjacent MTA can either serve these recipient(s) directly or may act as a relay on the path to the final destination. For each of these relaying or destination MTAs, a separate copy of the message is generated and handed over to the Reliable Transfer Server (RTS) for transmission to the adjacent MTA.

The RTS, in turn, sets up a session connection to its peer and reliably transfers the message as an indivisible unit, automatically recovering from possible transmission errors, nonpermanent line failures, and node failures. The RTS bases its service on a subset of the ISO session service, the Basic Activity Subset (BAS), which in turn makes use of ISO transport connections. In the ENC prototype, these transport layer connections employ X.25 switched virtual cir-

Figure 9 Architecture and virtual machines of the ENC PROFS / X.400 prototype

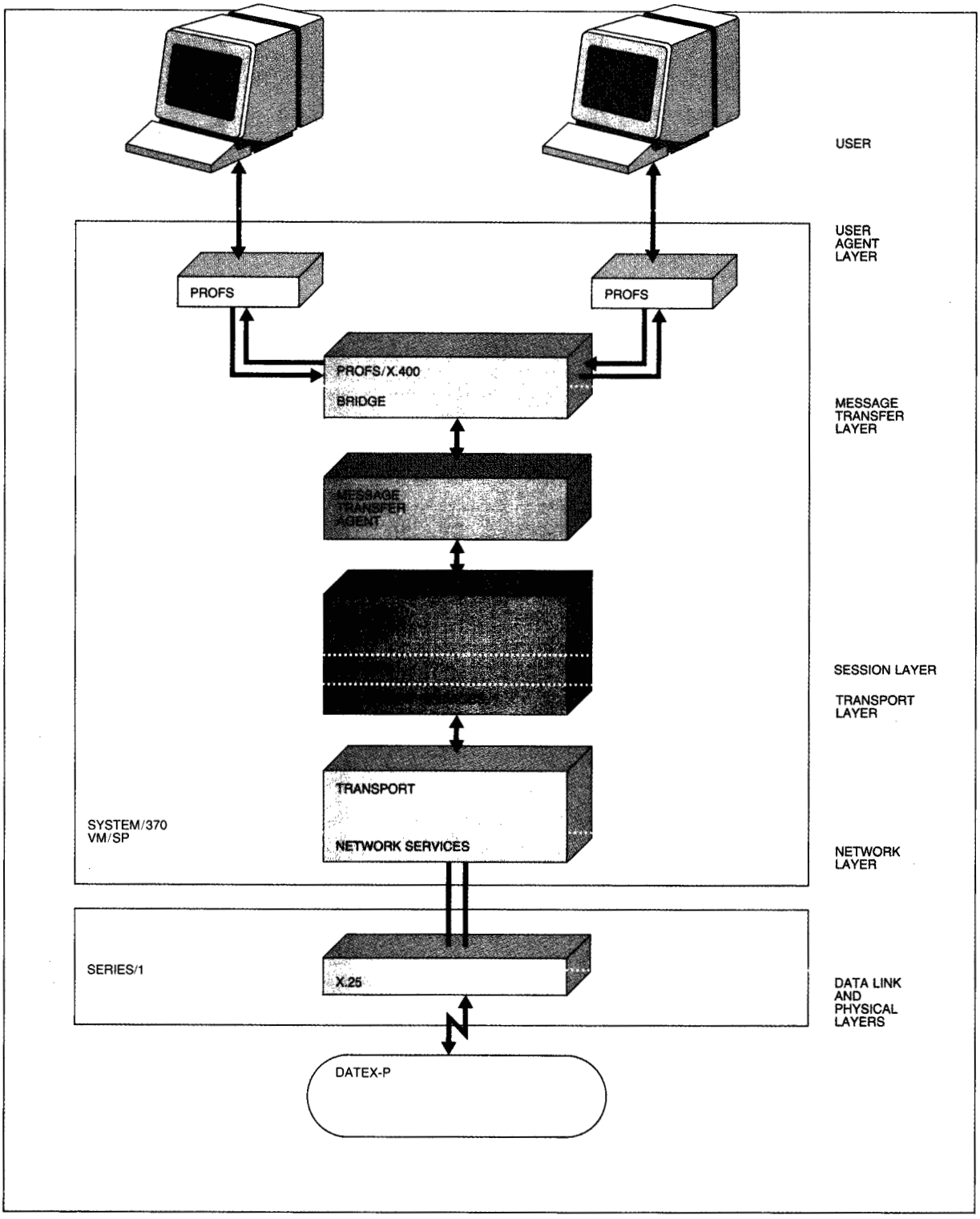


Figure 10 Communication between virtual machines

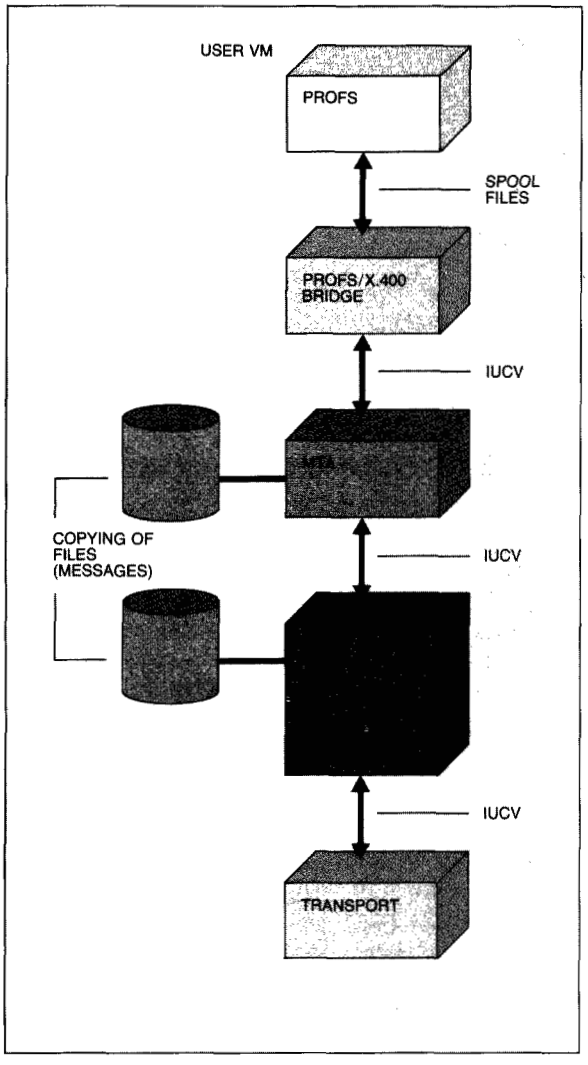
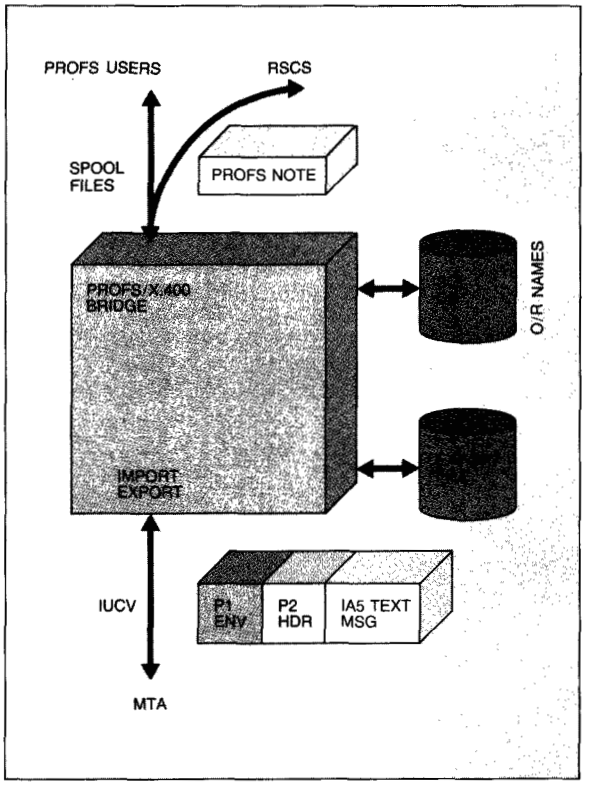


Figure 11 Design of the PROFS/X.400 bridge



cuit connections, which in Europe are provided by the national PTTs.

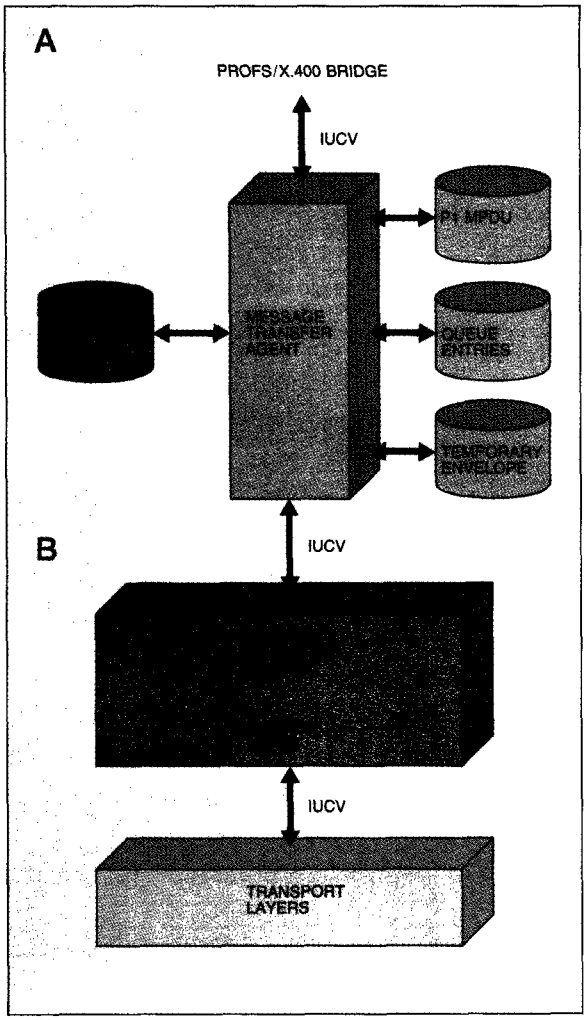
All of the functional components discussed above have been implemented as separate virtual machines. The virtual machines with the functions implemented in each are depicted in Figure 9, which also relates the functional components to the OSI layers.

Figure 10 provides an overview of the communications mechanisms employed between the virtual machines. The PROFS user agent, which is executing in the user's virtual machine, spools its output (PROFS notes) to the PROFS/X.400 bridge. The PROFS/X.400 bridge interfaces to the MTA via the Inter-User Communication Vehicle (IUCV), which is the standard communication mechanism of VM/SP. In addition, both the MTA and RTS virtual machines use a shared disk to provide safe storage for the messages.

All of the functional components have been implemented as separate virtual machines.

Figure 11 provides further details of the PROFS/X.400 bridge, which is the entry point for VM users to the

Figure 12 (A) Design of the Message Transfer Agent (MTA); (B) Reliable Transfer Server (RTS), session, and transport layers



X.400 network. Above is shown the path for incoming messages for local users or remote users wishing to access the X.400 network from an RSCS-based network (e.g., IBM's VNET). For these users, a PROFS note is automatically spooled to the PROFS/X.400 bridge. For all X.400 recipients of the note, the bridge employs an SQL database to map the PROFS-specific address information (userid,nodeid) to a valid O/R name. Two copies of the message are generated: one for recipients on the RSCS system and the other for X.400 recipients. The first set of messages is spooled to the RSCS virtual machine for distribution, and the second is forwarded to the X.400 MTA via IUCV.

Figure 12A shows the structure of the MTA, which maps the logical O/R names to the physical addresses of the next MTAs to receive the message. This is accomplished via a look-up in a local routing table.

To minimize communication costs, only one copy of the message is generated out of a single input message.

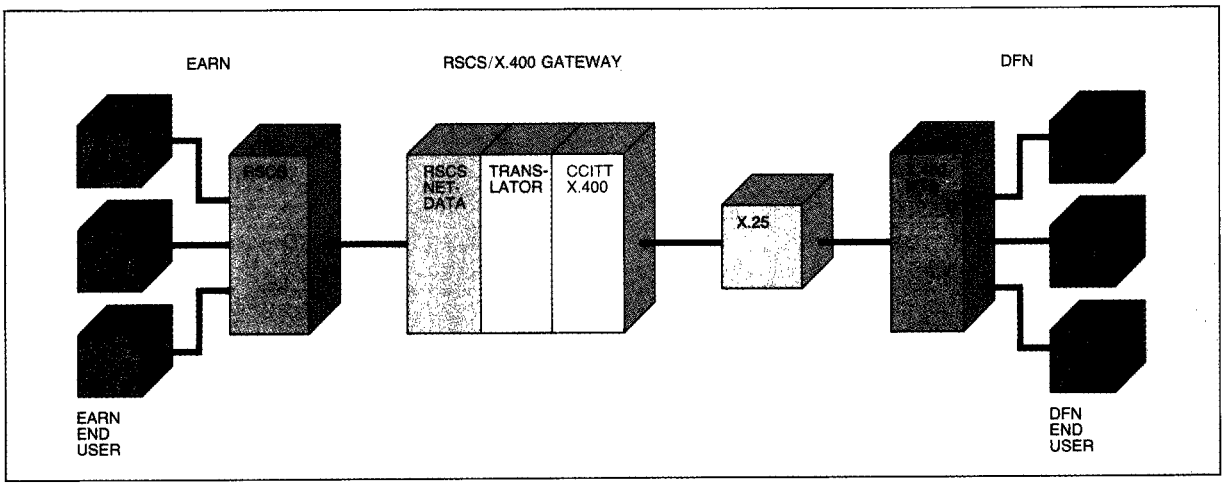
Temporary envelopes are produced for each recipient of the message and stored in a file. Finally, the complete P1 protocol data unit, including the necessary control information plus the original content of the note, is generated. To minimize the communication costs, only one copy of the message is generated out of a single input message for all recipients whose O/R names resolve to the same physical destination.

When the RTS receives the message, it assumes responsibility for transferring the message to its peer RTSs. In order to complete the transfer, the RTS invokes the session services, which in return result in calls to the transport services to transfer the message to the peer RTSs. The RTS, session layer, and transport layer are shown in Figure 12B.

The CCITT recommendation on the RTS (X.410) states that only transport protocol class 0 is to be provided by all X.400 message-handling implementations as a mandatory feature. Other transport protocol classes are either optional or for future study. To allow for the integration of alternative transport protocol implementations, the ISO transport service interface has been implemented in the ENC prototype, with the transport layer itself executing in a separate virtual machine.

The actual interface to the X.25 network is realized by a Series/1 processor that is channel-attached to the System/370 system running the transport layer virtual machine. Existing Series/1 software (PRPQ)

Figure 13 Conceptual location of a gateway



for X.25 protocols is supplemented with channel support code to effect packet transfer.

RSCS/X.400 gateway. The transport system of EARN and BITNET is the Remote Spooling Communications Subsystem (RSCS); it is the communication vehicle for a store-and-forward network. That is, the RSCS network accepts files from users, stores them, and transfers them whenever possible to the next host until the file reaches its final destination. For message handling, the NETDATA format is used. NETDATA may be considered as a protocol layer on top of RSCS and is responsible for the mail-specific data of any file. This includes such functions as the time the mail was sent, the name of the data file, and acknowledgments when received.

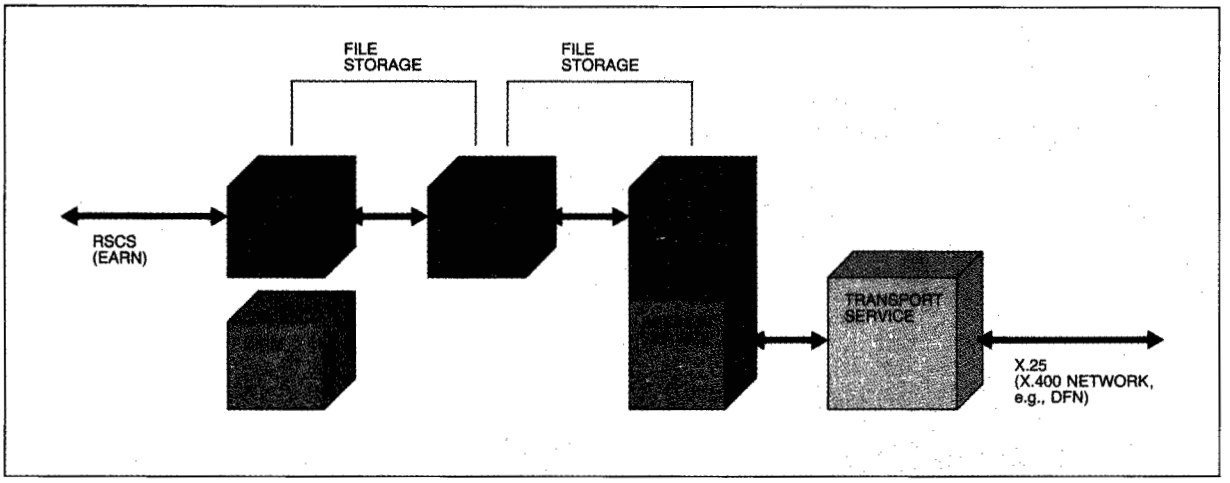
The commands NOTE400 and RECEIVE enable a remote user to make use of the RSCS/X.400 gateway. The NOTE400 command invokes an editor for the *ad hoc* preparation of short messages in the same manner as the CMS NOTE command. It assists the user in generating the control and address information required by the X.400 specifications and stores them into a message file that is internally structured in the same manner as a Teletex document. The messages generated by NOTE400 are forwarded to a gateway node via RSCS. A service process in the gateway node receives the messages and processes them as required. Messages may be received by an RSCS recipient via the standard CMS RECEIVE command.

The gateway allows the user-transparent transfer of messages between the EARN and BITNET networks and an X.400 network. Whereas RSCS is the message carrier whenever the gateway is accessed via EARN or BITNET, a packet-switched network with an X.25 protocol interface is used whenever messages coming from an X.400 network have to be forwarded to EARN and BITNET recipients. The configuration of the gateway is shown in Figure 13. With regard to the CCITT MHS model, the gateway behaves in the same manner as any MTA. In the case of RSCS, it behaves as any node that temporarily stores received messages until they can be forwarded either to an end user or to another node. Whenever forwarding is not possible, the originator of the message receives a negative acknowledgement.

Internal design. The gateway is implemented under the VM/SP operating system, as shown in Figure 14. On the left of Figure 14, the EARN network is accessed via RSCS. On the right is the X.400 network, accessed via X.25. The boxes in Figure 14 represent the virtual machines that comprise the gateway; they are defined as follows:

RIO — The *RSCS Input/Output* machine is responsible for communicating with the EARN network. In this respect, it performs both some of the MTA and IPM functions. When the RIO receives an IPM form message from EARN, it adds a P1 envelope and makes separate copies for recipients

Figure 14 Overall structure of an RSCS-to-X.400 gateway



with different addresses. The inverse of these operations takes place when a message from DFN is received for EARN. RIO is also responsible for creating any requested delivery notifications for messages that are transmitted from DFN to EARN.

MPM — In addition to the functions of a relaying MTA, the *Message Protocol Mapper* is responsible for the format conversions between the EARN internal message structure and the formats used within X.400. This conversion includes not only the encoding of the control information according to the X.409 encoding scheme but also any required content conversions [e.g., EBCDIC to International Alphabet No. 5 (IA5)].

RTS — The *Reliable Transfer Service* and the underlying service machines are identical to the respective components of the PROFS/X.400 system.

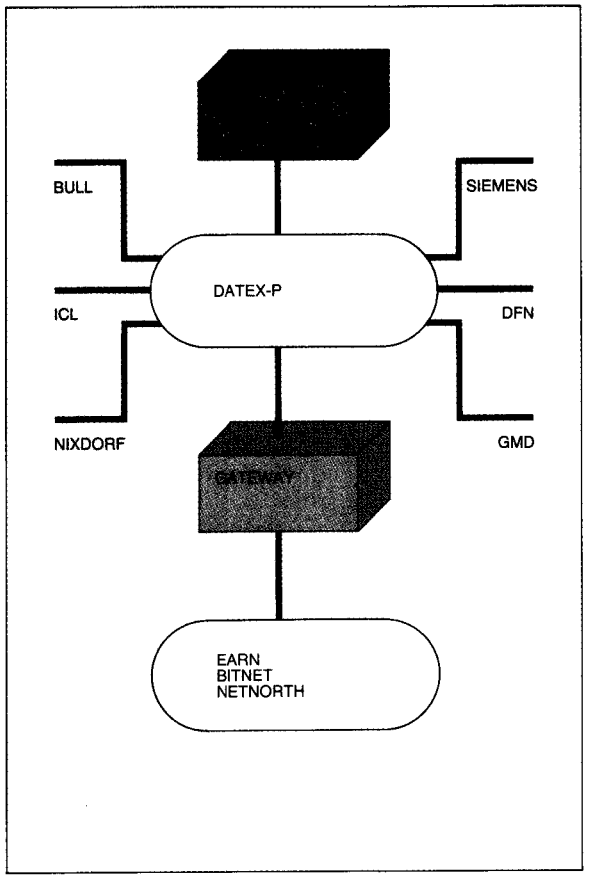
DIRM — The DIRM is the *Directory Manager* of all DFN subscribers who are known by the gateway. The directory contains routing and other delivery information for use by RIO and MPM.

Future enhancements. Because of the complexity and incompleteness of the X.400 recommendations, there were attempts almost immediately by a number of standards interpretation bodies to define consistent subsets that could be implemented. At this time, there exist two widely accepted proposals. The first is the subset defined by the CEN/CENELEC functional standard; the second is the CEPT profile specified by the association of the European PTTs.

Manufacturers participating with IBM in the demonstration included Siemens, Bull, ICL, and Nixdorf, as well as the scientific institution GMD and the DFN-Verein.

On the basis of this work, the European Research Networks department of the ENC is continuing to enhance the X.400 prototypes to support all parameters and options that will be required by these two standardization bodies. To complete this work, additional function at the user interface is required that was not available in the first version of the X.400 prototypes.

Figure 15 Summary of the X.400 cooperation for Fair Mail



In the PROFS/X.400 prototype, these functions can be integrated by the use of PROFS user exits made available in the latest release of PROFS. For the gateway, the NOTE400 user agent will be enhanced accordingly.

CeBIT Fair 1986. The PROFS/X.400 system and the RSCS/X.400 gateway were both demonstrated at the CeBIT Fair 1986 in Hannover, Germany. Manufacturers participating with IBM in the demonstration included Siemens, Bull, ICL, and Nixdorf, as well as the scientific institution GMD and the DFN-Verein.

Figure 15 is an overview of the Fair Mail systems that were part of the demonstration. Communication among individual systems was accomplished over the Datex-P network, which is the X.25 packet-switching network service offered by the Deutsche Bundespost (German PTT). In addition to communication between the X.400 systems via the X.25 network, the gateway enabled the X.400 users to communicate with all users in the EARN, BITNET, and NETNORTH[22] networks and vice versa.

Within the framework of Fair Mail, not all participating MTAs communicated directly with one another, as it is possible to do using the Datex-P network. Instead, those MTAs not communicating directly with one another exchanged messages through the use of intermediate or relaying MTAs, thus demonstrating the relaying of messages.

During several meetings prior to the demonstration between the participants, an X.400 profile for Fair Mail was established. This profile defined the details and arrangements that were required to unify the participating systems, in order to allow communication among them. The profile contains all aspects that are not clearly defined in the X.400 recommendations, such as the requirements for support of optional service elements, the architecture of O/R-names, and the types of user-provided data that were to be exchanged among the participating systems. The Guide to the Use of Standards, Profile A/3211, Version 2.0 of the Standards Promotion and Application Group (SPAG), was used as a basis for the X.400 profile for the Fair Mail.

Future standards efforts

The CCITT X.400 recommendations can be considered as a milestone on the way to a globally accepted message-handling system. To date, there are four major ongoing standards efforts initiated by the recommendations.

First, there are attempts under way to streamline the existing X.400 functions. While embracing the standard as a whole, at the same time the aim is to remove seldom-used features that introduce additional complexity with no apparent improvement in usability or function. The CEPT and CEN/CENELEC functional standards can be considered as an activity to promote implementations while maintaining compatibility among different manufacturers' systems.

The second major effort is to define a suitable directory service to complete the X.400 MHS model. The current recommendations implicitly assume the availability of a directory service, the details of which were left for further study in the 1984 version of the recommendations. Because a directory service is essential to the operation of a worldwide X.400 MHS,

the CCITT itself has continued this work. It is possible that stable versions of the draft recommendations on directory services[23] will be available by the end of 1987.

The third activity is to refine the definition of the general structure and service interfaces of the MHS model. The ISO Message Oriented Text Interchange System (MOTIS) draft standards[13-16] define a cleaner interface between the user-agent sublayer (UASL) and the message-transfer sublayer (MTSL) to get the interpersonal messaging specific parameters out of the message transfer part. This makes the MTSL look like an application-independent store-and-forward network that is not restricted to personal messaging but could also be used for yet-to-be-defined ISO applications.

In addition to this refinement, the ISO draft incorporates additional management functions that are required to operate an MHS network consisting of a set of privately operated cooperating MTAs without the interference (and assistance) of an administration management domain.

The fourth and last set of activities triggered by the X.400 recommendations are a number of research projects, particularly in the area of group communication. Many of them are urgently required to shed some light on the nontechnical problems of user acceptance of computer-based message-handling systems. Security and data privacy issues are also being considered.

Concluding remarks

This paper has presented the X.400 recommendations and the work toward implementing those recommendations in the form of two prototypes at the IBM European Networking Center in Heidelberg, Germany. The prototypes have greatly assisted the understanding of these standards and how they can be integrated into the IBM office products.

There has been much progress toward the standardization of message-handling systems via the X.400 recommendations. However, much work still remains, particularly in the areas of directory architecture and the exchange of more complex documents via electronic mail systems. IBM office architectures, such as SNADS, DIA, and DCA[24-26] today more completely address the interchange of memos and other documents in the office environment. Thus they can help in pointing the way for future standards efforts in this area. The X.400 recommendations are, however, an important first step in achieving information exchange between computers of different manufacturers.

Acknowledgments

The authors offer special thanks to Dr. Günter Müller, Director of the IBM European Networking Center, for his leadership, inspiration, and encouragement. Under his direction, the IBM European Networking Center was founded in Heidelberg for research into telecommunications standards, and the prototypes were developed. The authors also thank Werner Schulz, Dietrich Kropp, and Roger Holliday for their work in developing the prototypes as members of the ENC. Finally, we would like to thank Günter Schulze, Horst Ehmke, and Ursula Viebeg from the GMD, and Andy Hooper from Queen's University, Kingston, Ontario, Canada, each of whom contributed greatly in the design, coding, testing, and demonstration of the X.400 prototypes.

Cited references and notes

1. P. C. Gardner, Jr., "A system for the automated office environment," *IBM Systems Journal* **20**, No. 3, 321–345 (1981).
2. *Red Book, Volume VIII.7, Data Communication Networks Message Handling Systems*, CCITT Eighth Plenary Assembly, Malaga-Torremolinos, Spain (October 1984).
3. The standards associations that are involved in this effort include CEN (Comité Européen de Normalisation), CENELEC (Comité Européen de Normalisation Electrique), CEPT (Conférence Européenne des Administrations des Postes et des Télécommunications), SPAG (Standards Promotion and Application Group), NBS (National Bureau of Standards in the U.S.), and COS (Corporation for Open Systems).
4. The participants at CeBIT 86 were IBM, Siemens, Nixdorf, Bull, ICL, Gesellschaft für Mathematik und Datenverarbeitung (GMD), and Verein zur Förderung eines Deutsches Forschungsnetzes (DFN-Verein).
5. J. D. Day and H. Zimmermann, "The OSI Reference Model," *Proceedings of the IEEE* **71**, No. 12, pp. 1331–1333 (December 1983).
6. Other OSI applications include FTAM for file transfer and VTP for virtual terminal, although to date these standards are still under development and have not been widely implemented.
7. *Information Processing Systems—Open Systems Interconnection—Basic Connection Oriented Session Service Definition*, ISO International Standard 8326 (September 1984).
8. *Information Processing Systems—Open Systems Interconnection—Transport Service Definition*, ISO International Standard 8072 (June 1986).
9. *Red Book, Volume VIII.5, Recommendation X.214: Transport Service Definition for Open Systems Interconnection (OSI) for CCITT Applications*, CCITT Eighth Plenary Assembly, Malaga-Torremolinos, Spain (October 1984).

10. *Red Book, Volume VIII.5, Recommendation X.224: Transport Protocol Specification for Open Systems Interconnection (OSI) for CCITT Applications*, CCITT Eighth Plenary Assembly, Malaga-Torremolinos, Spain (October 1984).
11. *Information Processing Systems—Open Systems Interconnection—Connection Oriented Transport Protocol Specification*, ISO International Standard 8073 (July 1986).
12. *Red Book, Volume VIII.6, Recommendation X.300: General Principles and Arrangements for Interworking Between Public Data Networks, and Between Public Data Networks and Other Public Networks*, CCITT Eighth Plenary Assembly, Malaga-Torremolinos, Spain (October 1984).
13. *Information Processing—Text Communication—Functional Description and Service Specification for Message Oriented Text Interchange System*, ISO Draft International Standard 8505 (1986-02-27), available from the American National Standards Institute, 1430 Broadway, New York, NY 10018.
14. *Information Processing—Text Communication—Message Oriented Text Interchange System, Message Transfer Sublayer, Message Interchange Service and Message Transfer Protocol*, ISO Draft International Standard 8883 (1986-02-27), available from the American National Standards Institute, 1430 Broadway, New York, NY 10018.
15. *Information Processing—Text Communication—Message Oriented Text Interchange System User Agent Sublayer—Interpersonal Messaging User Agent—Message Interchange Formats and Protocols*, ISO Draft Proposal 9065 (1985-12-02), available from the American National Standards Institute, 1430 Broadway, New York, NY 10018.
16. *Information Processing—Text Communication—Message Oriented Text Interchange System—Reliable Transfer Server and Use of Presentation and Session Services*, ISO Draft Proposal 9066 (1985-12-02), available from the American National Standards Institute, 1430 Broadway, New York, NY 10018.
17. In the X.400 recommendation on "System Model—Service Elements" an even more abstract way of describing the Message Transfer and Interpersonal Messaging Services is introduced. This makes use of service elements, which are synonyms for service specifications in plain English text, without any formalism or relation to a possible implementation of the service interfaces. There is not a one-to-one mapping between service elements and service primitives. On one hand, a service primitive and its parameters can support several service elements, and on the other hand, several service primitives may be specified to support one service element.
18. *DFN—Deutsches Forschungsnetz (German Research Network), The DFN Message Handling System—Specifications for Realization*, DFN Central Project Office, Berlin (March 1985).
19. *Remote Spooling Communications Subsystem Networking: Program Reference and Operations Manual*, SH24-5005-2, IBM Corporation; available through IBM branch offices.
20. The Remote Spooling Communication Subsystem (RSCS) is a set of protocols defining a store-and-forward file-transfer network that is based on leased lines. VNET is the IBM internal data communications network and is used for file transfer and interpersonal messaging. The European Academic and Research Network (EARN) is the European branch of the international BITNET (Because It's Time Network) that has been established by universities and research institutes to provide for electronic mail and file transfer in the scientific world.
21. J. S. Quateman and J. C. Hoskins, "Notable computer networks," *Communications of the ACM* **29,** No. 10, 932–971 (October 1986).
22. G. Müller, R. Holliday, and G. Schulze, "A message-handling gateway between EARN/BITNET and DFN," *Proceedings of the IFIP TC 6 International Symposium on Computer Message Systems*, Washington, DC, September 5–7, 1985, pp. 157–170.
23. *X.ds Series of Draft Recommendations for Directory Systems, Version 4*, Melbourne, Australia (April 1986).
    X.ds1 The Directory—Overview of Concepts, Models, and Services
    X.ds2 The Directory—Information Framework
    X.ds3 The Directory—Access and System Protocols
    X.ds4 The Directory—Selected Attribute Types
    X.ds6 The Directory—Selected Object Classes
    X.ds7 The Directory—Authentication Framework
    X.ds8 The Directory—Abstract Service Definition
    X.ds9 The Directory—Procedures for Distributed Operation
24. B. C. Housel, Jr., and C. J. Scopinich, "SNA Distribution Services," *IBM Systems Journal* **22,** No. 4, 319–343 (1983).
25. T. Schick and R. F. Brockish, "The Document Interchange Architecture: A member of a family of architectures in the SNA environment," *IBM Systems Journal* **21,** No. 2, 220–244 (1982).
26. M. R. DeSousa, "Electronic information interchange in an office environment," *IBM Systems Journal* **20,** No. 1, 4–22 (1981).

Thomas E. Schütt *IBM European Networking Center, Tiergartenstrasse 15, D-6900 Heidelberg, West Germany.* In 1985, Dr. Schütt joined the IBM European Networking Center, where he participated in the design and development of the X.400 prototype message-handling systems. He is currently responsible for investigating methods of interconnecting local and wide-area networks. Dr. Schütt received a Diploma in computer science and economics from the University of Kiel in 1980 and a Doctorate in computer science from the Technical University of Munich in 1985. He is a member of the ACM and the German Gesellschaft für Informatik (GI).

James B. Staton III *IBM European Networking Center, Tiergartenstrasse 15, D-6900 Heidelberg, West Germany.* Mr. Staton joined the IBM Communication Products Division in 1978 at Research Triangle Park, North Carolina. He joined the Systems Network Architecture (SNA) group in 1981 and participated in several enhancements to SNA. Initially he worked on network management issues involving modems and communications links, for which he has a patent application. Mr. Staton participated in the design and development of Version 2 of the Teleprocessing Network Simulator (TPNS) program product, which simulates networks and devices for performance and reliability testing. More recently, he managed the communications architecture department and the standardization of communications protocols for the IBM Token-Ring Network and its integration into SNA. He is currently on assignment to the European Networking Center, where he manages the development of OSI prototype systems. Mr. Staton received the B.A. degree in mathematics from Guilford College, Greensboro, North Carolina, and the M.S. degree in computer science from the Ohio State University, Columbus, Ohio. He is a member of the IEEE.

Wilhelm F. Racke *IBM European Networking Center, Tiergartenstrasse 15, D-6900 Heidelberg, West Germany.* Dr. Racke joined

IBM Germany in 1984 at IBM Germany Headquarters in Stuttgart, where he worked in the area of office communication. In 1985 he transferred to the IBM European Networking Center, where he was the technical leader responsible for the development of the ENC X.400 demonstration at CeBIT 86 in Hannover. Dr. Racke is currently working in the area of directories for communication systems, where he participates in the DIN (German Standards Institute) committee for directories. Dr. Racke received his Diploma in mathematics and computer science from the University of Karlsruhe in 1977 and his Doctorate in computer science from the University of Kaiserslautern in 1984. He is a member of the ACM and the German Gesellschaft für Informatik (GI).

Reprint Order No. G231-5296.